Языки описания схем

mk.cs.msu.ru → Лекционные курсы → Языки описания схем

Блок K5

Кое-что ещё:
SPI для двух устройств

Лектор:
Подымов Владислав Васильевич
E-mail:
valdus@yandex.ru

ВМК МГУ, 2023/2024, осенний семестр

База SPI: общее описание

SPI (Serial Peripheral Interface) — это
самое простое семейство протоколов
синхронной последовательной передачи данных между устройствами

Это семейство дуплексных протоколов:
данные пересылаются одновременно в две стороны

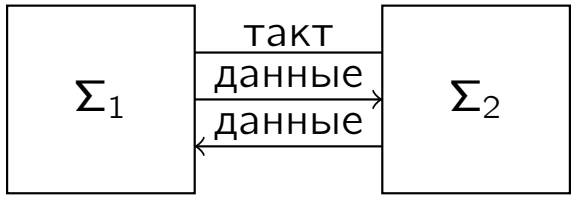
Начнём с самого простого (базового) варианта этого протокола:
попробуем организовать синхронную передачу чисел ширины 8
между двумя устройствами

Дуплексность: для пересылки данных потребуются два провода

Синхронность: для пересылки часов потребуется *хотя бы* один провод

Остановимся на трёх проводах

База SPI: общее описание



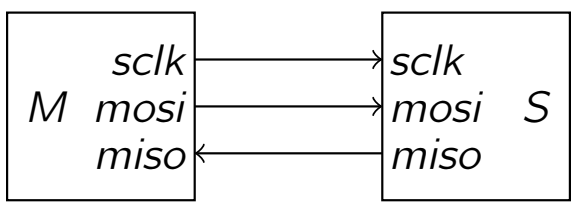
Кто кому должен переслать тактовый сигнал?

Заранее выдадим роли устройствам:
одно — ведущее (master; *M*), другое — ведо́мое (slave; *S*)

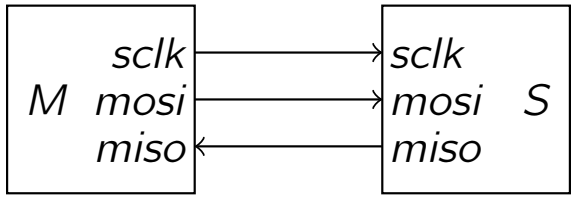
Ведущее устройство пересылает тактовый сигнал ведо́мому

Общепринятые названия портов (на картинке — сверху вниз):

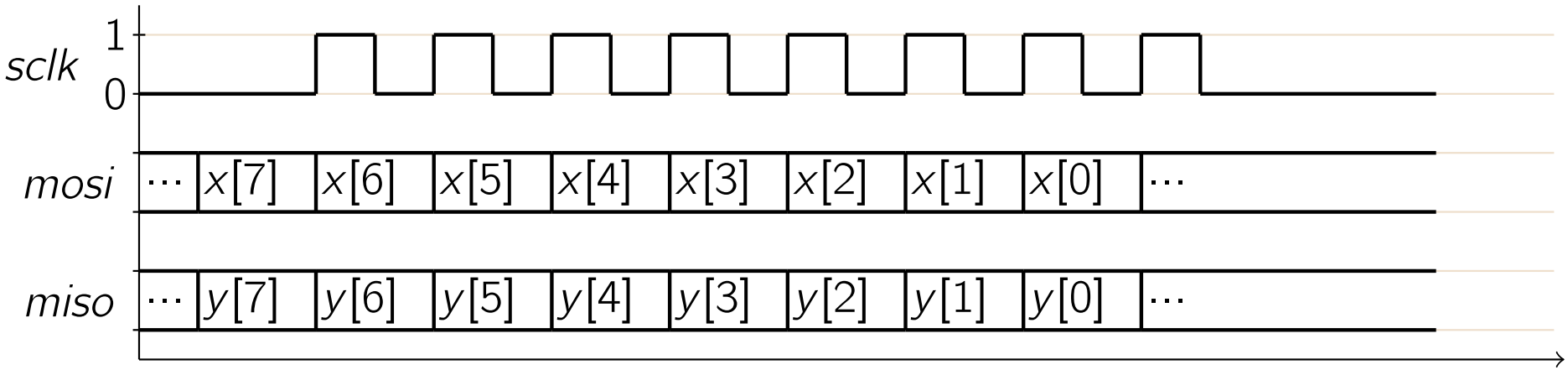
- sclk (serial clock)
- mosi (master out slave in; выход ведущего, вход ведомого)
- miso (master in slave out; вход ведущего, выход ведомого)



База SPI: общее описание



Сразу к делу: пересылка сообщений x (от M к S) и y (от S к M) ширины 8 в «базовом» варианте протокола



M решает, проводить ли передачу

Передачи нет $\Rightarrow$ *sclk* не осциллирует (тишина)

Передача есть $\Rightarrow$

- *sclk* осциллирует 8 раз
- Передним фронтом *sclk* отмечается начало передачи следующего разряда

База SPI: общее описание

Другие варианты передачи между двумя устройствами

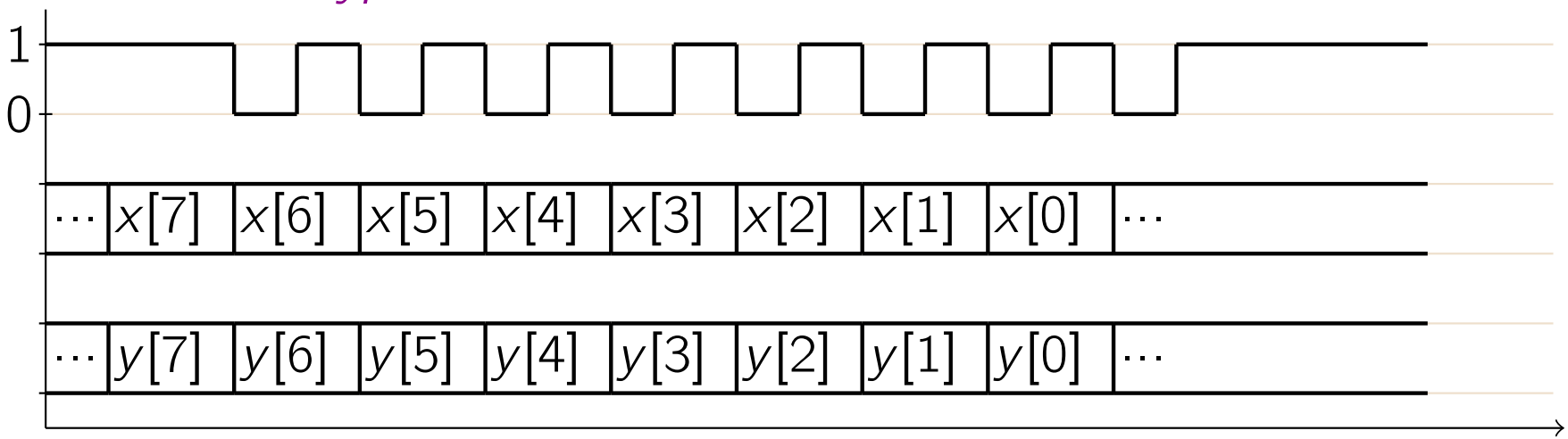
Другая ширина сообщения ⇒ соответствующее количество осцилляций (*но число битов должно быть оговорено заранее*)

- *Самая популярная ширина* — 8

Другой порядок пересылки — изменения в протоколе очевидны

- *Самый популярный порядок — от старшего бита к младшему*

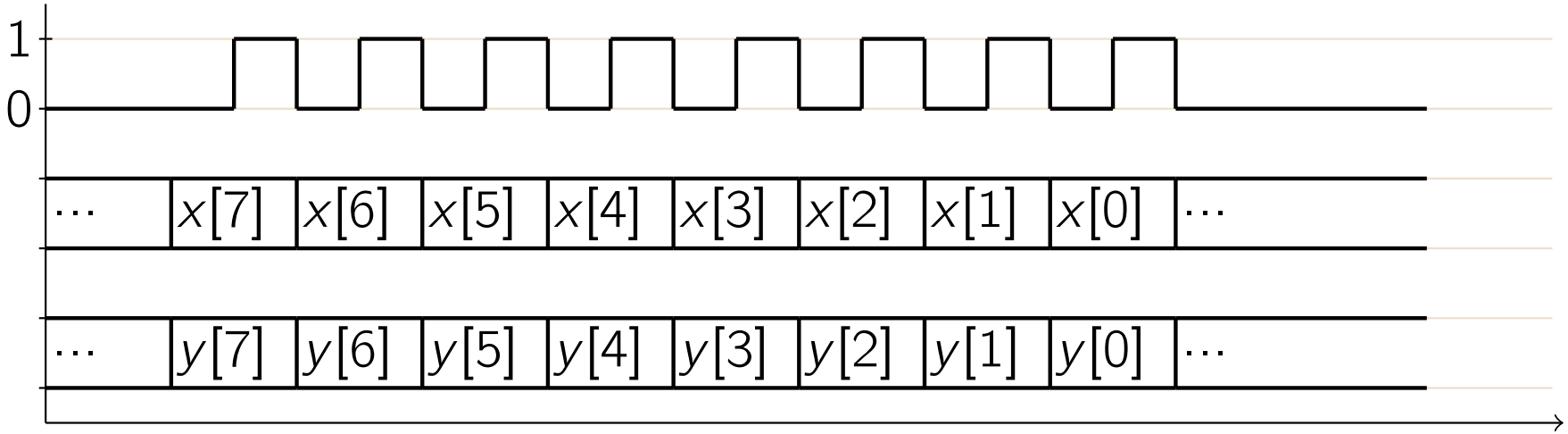
Другой активный уровень сигнала sclk:



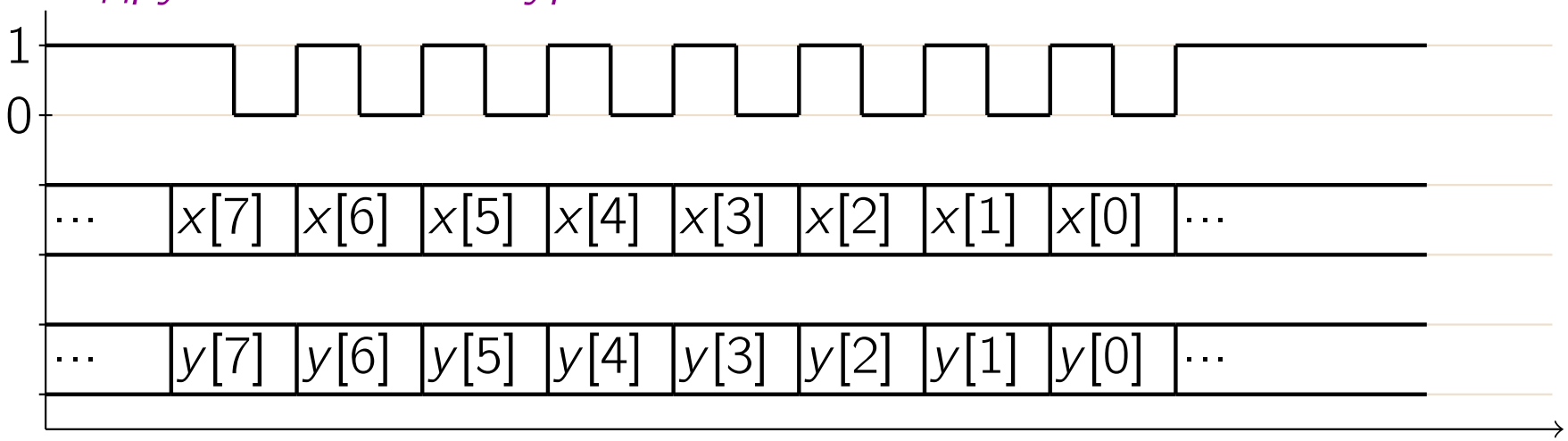
База SPI: общее описание

Другие варианты передачи между двумя устройствами

Смещение данных на половину такта



<...> и другой активный уровень sclk:



База SPI: реализация

В реализации UART
приёмник был устроен заметно сложнее передатчика:

- Передатчик пересылает биты сообщения согласно протоколу,
  не думая о погрешностях и том, как сообщение будет приниматься
- Приёмник должен корректно принять сообщение
  для любых (*разумных*) частоты, фазы и погрешностей передатчика,
  отталкиваясь только от частоты протокола

Высокая сложность приёмника по сравнению с передатчиком —
это одна из характерных черт асинхронных протоколов

В простом базовом варианте SPI (*и в других синхронных протоколах*)
схема приёмника обычно устроена не сложнее схемы передатчика

Далее будем *полагать*, что оба устройства
знают время начала и конца передачи каждого сообщения

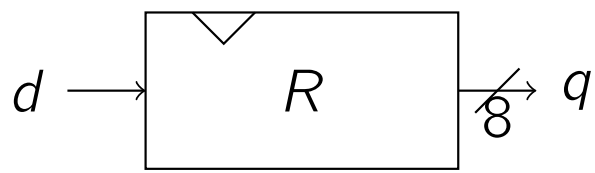
База SPI: реализация приёмника

Всё, что нужно уметь делать при приёме сообщения, —

- принимать очередной бит
  по переднему фронту тактового сигнала протокола и
- выдавать последние 8 принятых битов

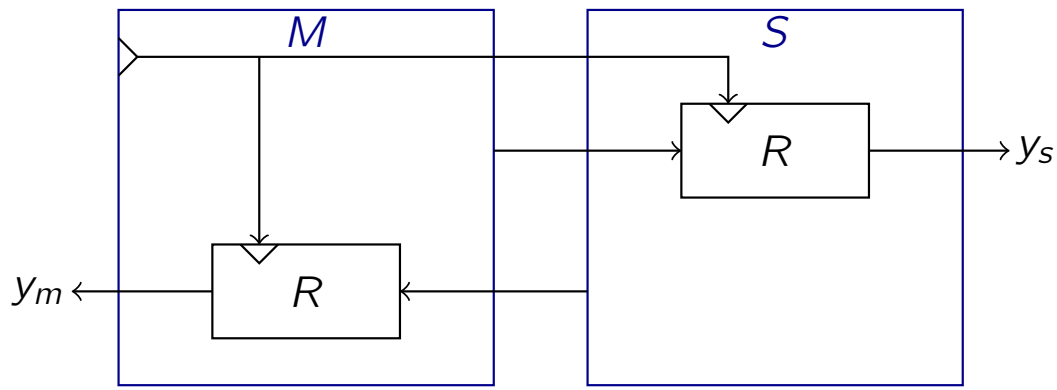
Это умеет делать сдвиговый регистр R ширины 8 такого вида:

если $q(t) = (q_7\, q_6 \ldots q_0)$, то $q(t+1) = (q_6\, q_5 \ldots q_0\, d(t))$



База SPI: реализация приёмника

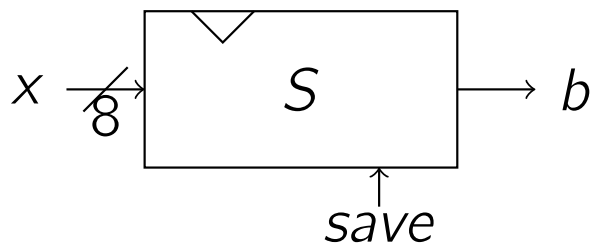
Устройство SPI-приёмников, направляющих принятые сообщения в соответствующие выходы y_m, y_s:



База SPI: реализация передатчика

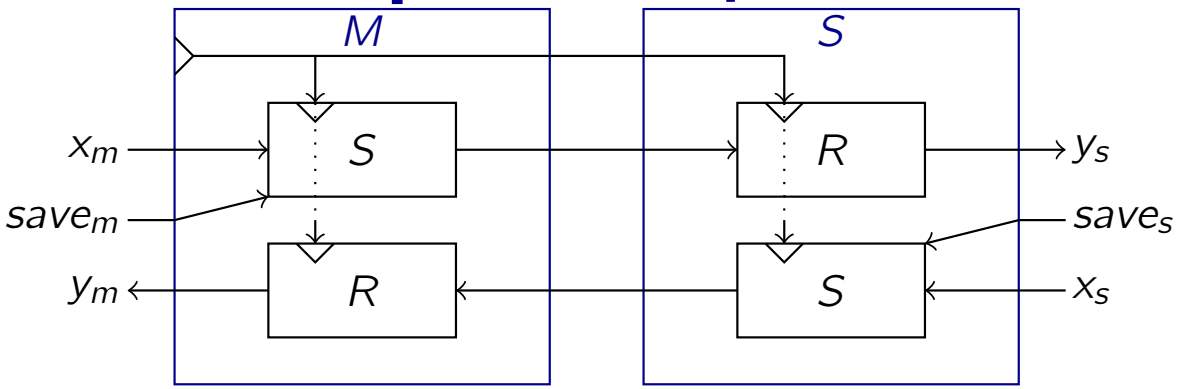
SPI-передатчик можно устроить так:

- согласно управляющему сигналу на входе (*save*) он сохраняет сообщение (x) и направляет на выход (b) старший разряд этого сообщения
- по каждому переднему фронту тактового сигнала он переходит к выдаче следующего разряда



Схемы с похожим поведением уже не один раз упоминались в курсе: это сериализаторы

База SPI: итоговая реализация

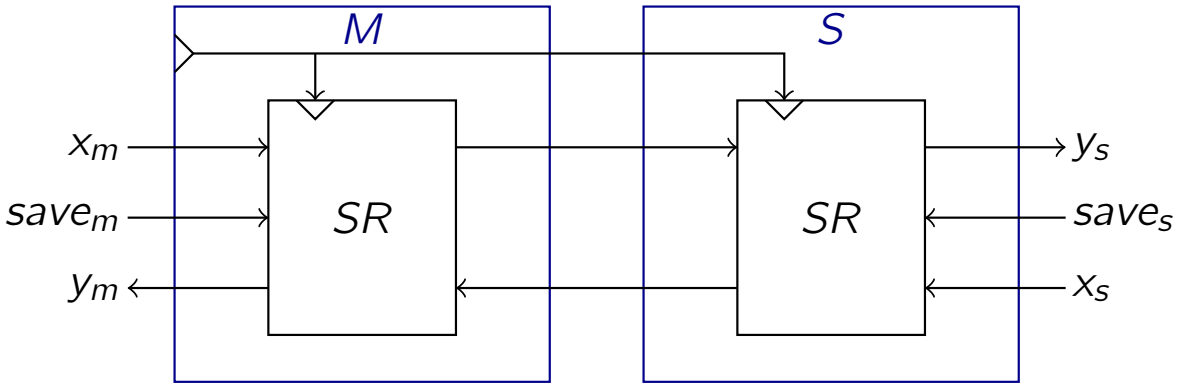


Подсхемы S и R имеют схожее поведение, и в связи с этим их можно объединить в единую подсхему SR — синхронный регистр, умеющий

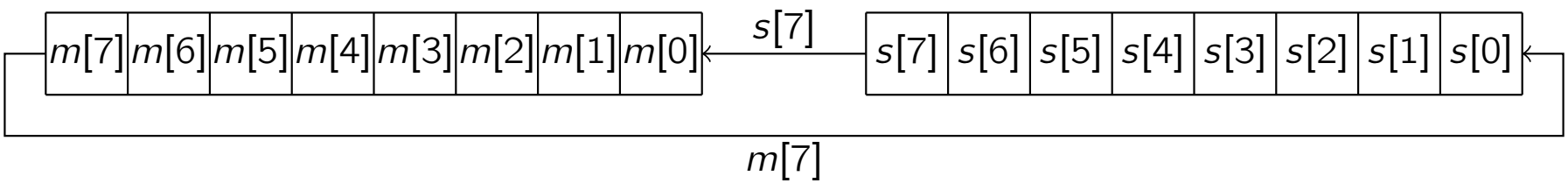
- хранить число ширины 8
- выдавать хранящееся число и его старший разряд
- сохранять число со входной шины
  (*как это делает параллельный регистр*)
- сдвигать число в сторону старшего бита,
  заполняя младший бит значением на одноразрядном входе
  (*как это делает сдвиговый регистр*)

После такого объединения в SR оказываются «перемешаны» принятые биты, и биты, которые осталось отправить

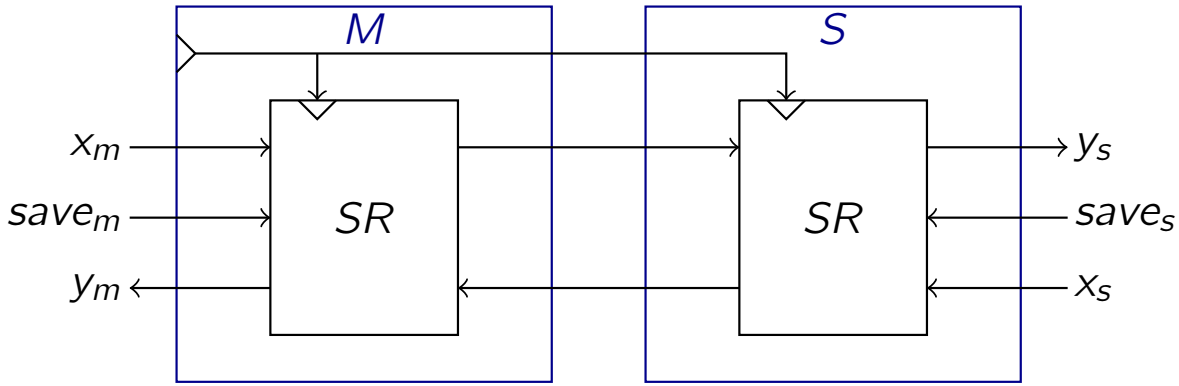
База SPI: итоговая реализация



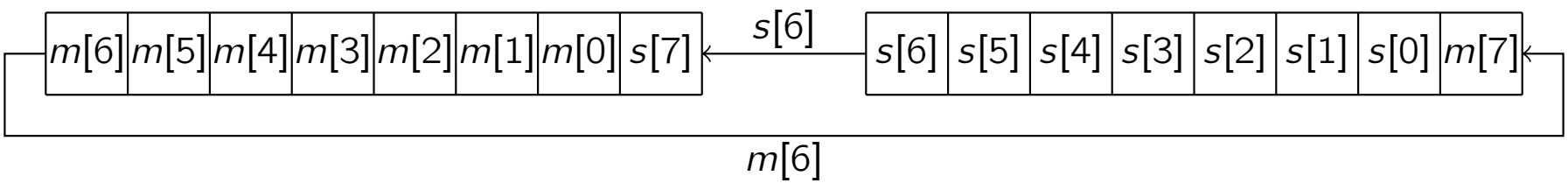
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



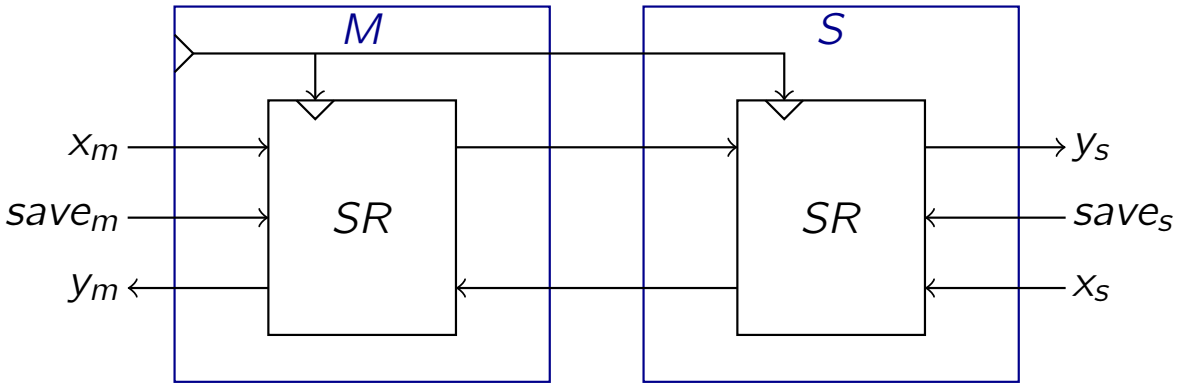
База SPI: итоговая реализация



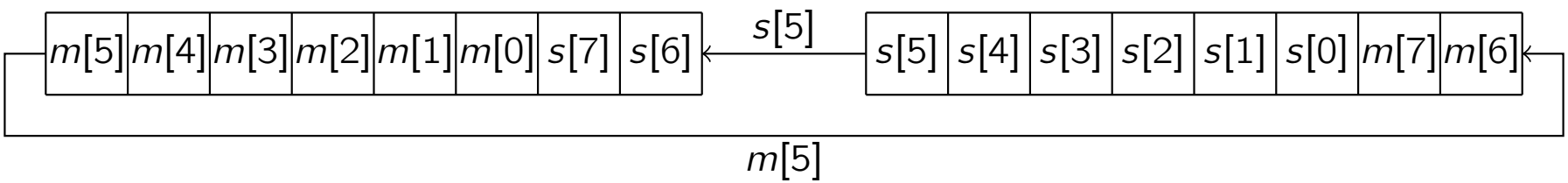
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



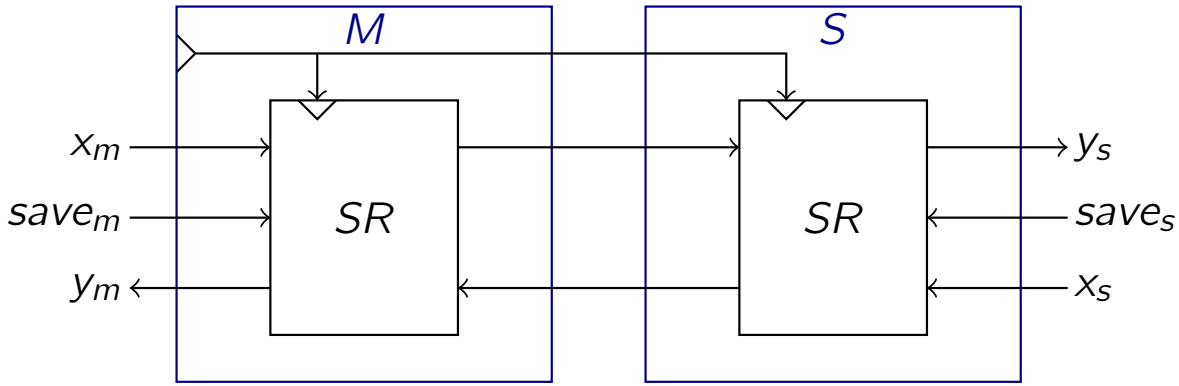
База SPI: итоговая реализация



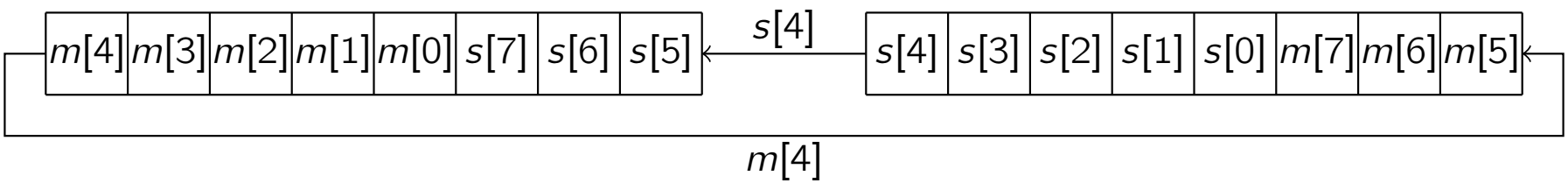
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



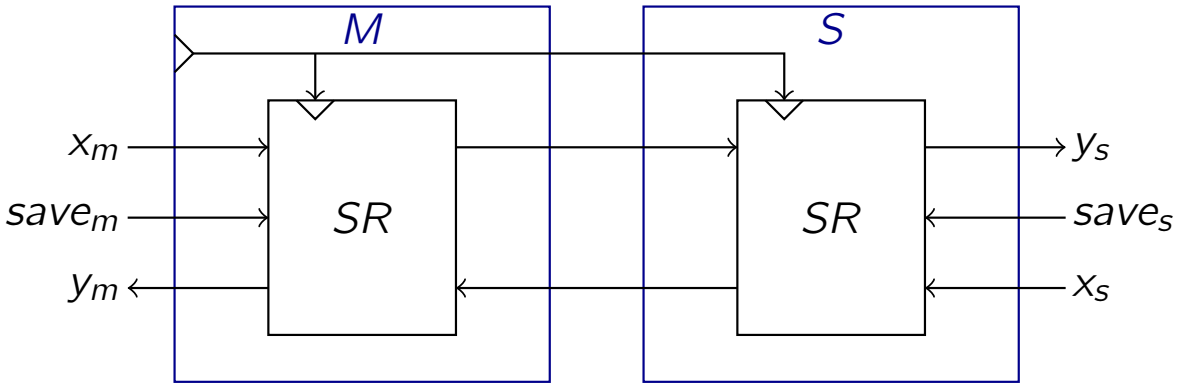
База SPI: итоговая реализация



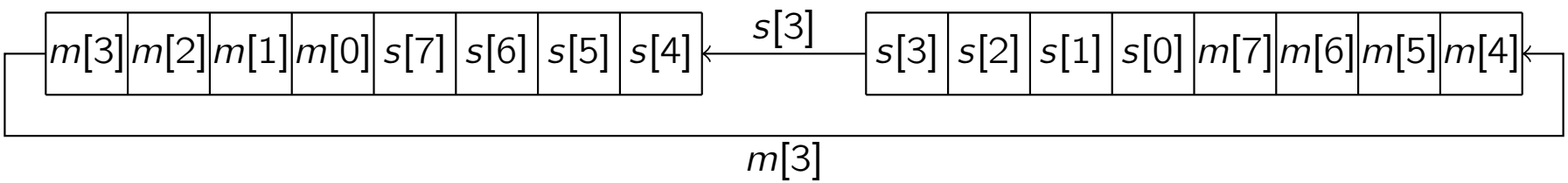
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



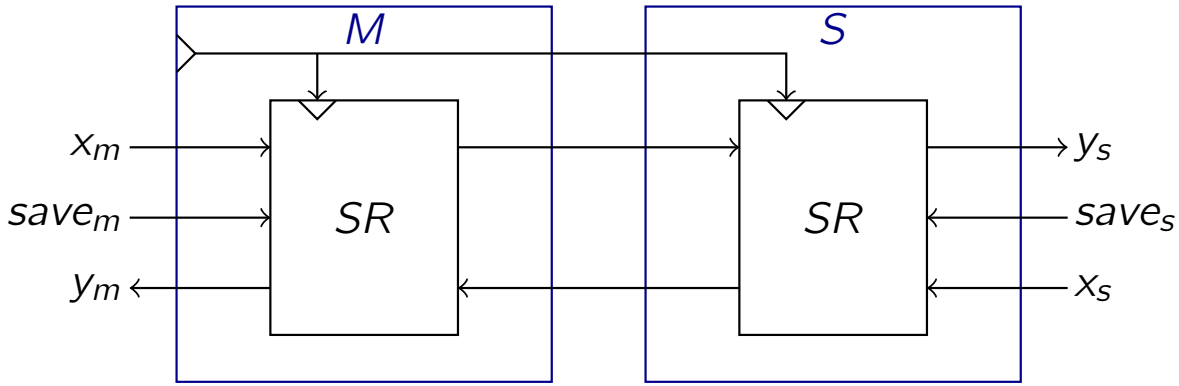
База SPI: итоговая реализация



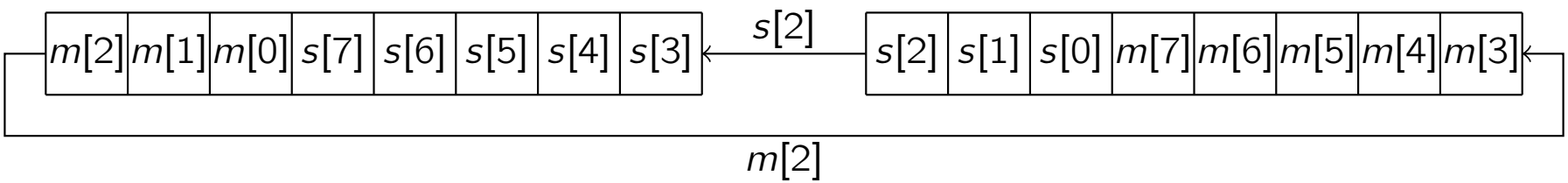
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



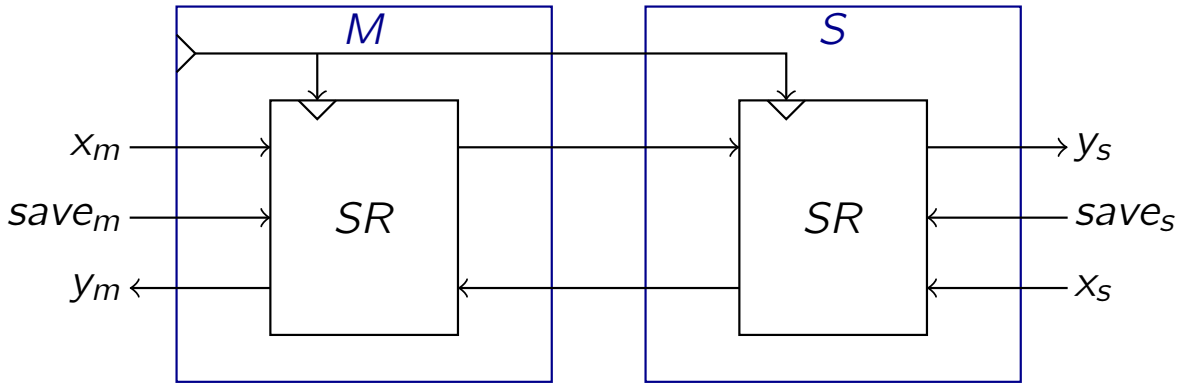
База SPI: итоговая реализация



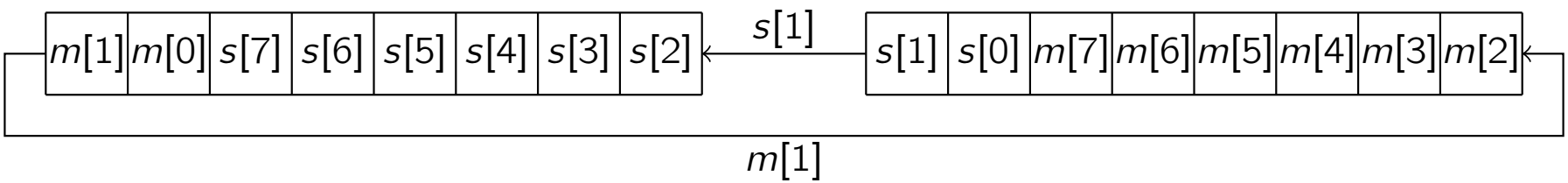
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



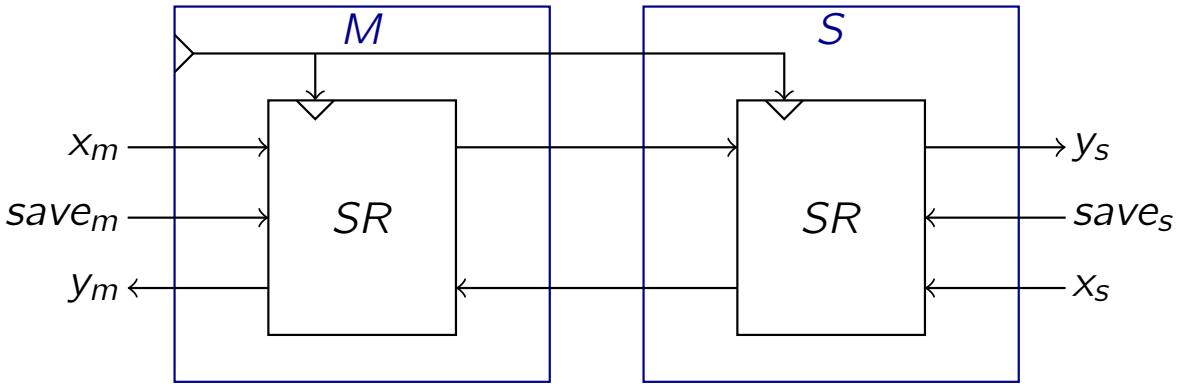
База SPI: итоговая реализация



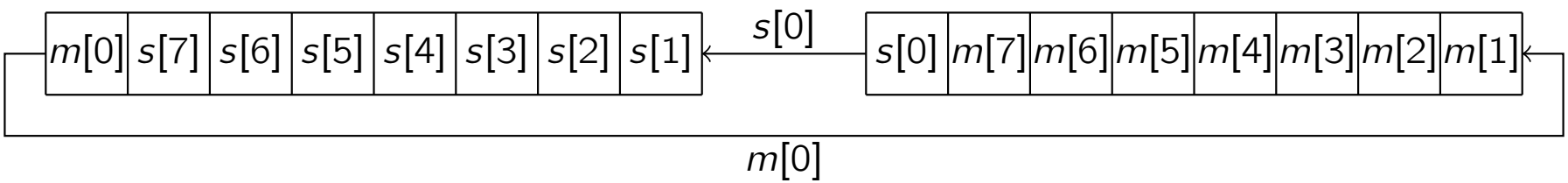
Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



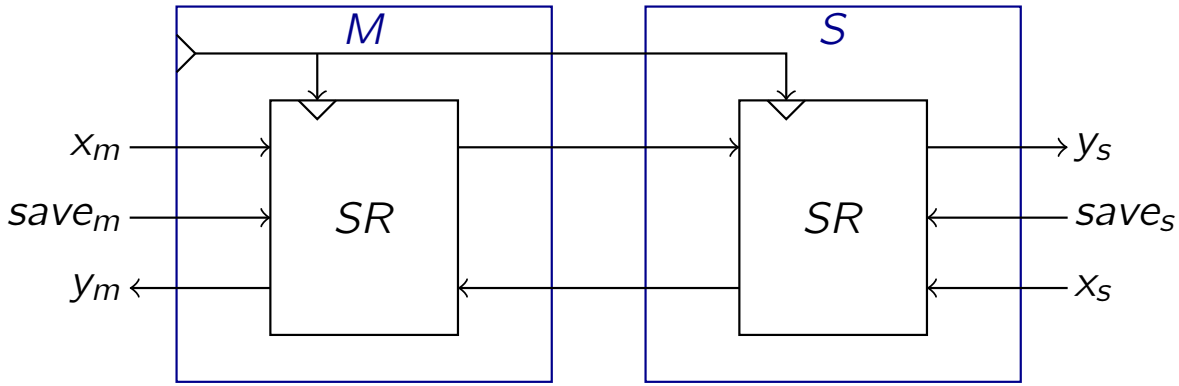
База SPI: итоговая реализация



Пересылка одной пары сообщений для SR по тактам протокола выглядит так:



База SPI: итоговая реализация



Пересылка одной пары сообщений для SR по тактам протокола выглядит так: